AD-A267 048

CUMENTATION PAGE

Form Approved
OMB No 0704-0188

2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Final Report 01 May 92 - 30 Apr 93

4. TITLE AND SUBTITLE
Microwave interaction with plasmas

5. FUNDING NUMBERS
F49620-92-J-0266

6. AUTHOR(S)
Drs Igor Alexeff and Mark Radar

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Tennessee
409 Feris Hall
Knoxville TX 37996-2100

8. PERFORMING ORGANIZATION REPORT NUMBER
AFOSR-TR- 93 0526

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NE
110 DUNCAN AVENUE SUITE B115
BOLLING AFB DC 20332-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
2301/ES

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
UNLIMITED

This document has been approved for public release and sale; its distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

SEE REPORT FOR ABSTRACT

93-16708

93 7 [illegible] 052

14. SUBJECT TERMS	15. NUMBER OF PAGES 14
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASS	UNCLASS	UNCLASS	UL

Final Report

To

AFOSR

Dr. Robert J. Barker
Technical Contract Monitor

From

Igor Alexeff and Mark Rader
409 Feris Hall
University of Tennessee
Knoxville, TN 37996-2100

Plasma Source Development

The object of this work is to create dense plasmas for research military, and industrial development that are accessable - and not confined in vacuum systems. The corona method of production by high D.C. and A.C. voltages is simple and convenient. Much work was done in this field in the 1900-1940 time frame, but the field seems to be neglected now.

Multiple points were used on positive and negative electrodes at voltages of 6000 to 1200 volts D.C. The current appears to comprise equal negative and positive ion species, as an metal plate blocking the current flow reduced the current exactly to 1/2 of its initial value.

An electrostatic voltmeter (zero current drain) was used to obtain the potential profiles.

Knowing the potential profiles, we obtained the electrostatic field at each point in space. Knowing the mobility, we obtained the ion drift velocity. Knowing the ion drift velocity, and the local current density, we found the ion density, which was remarkably high $10^{16}/m^3$ on the average.

We cross-checked the ion drift velocity by using a wind tunnel. The ion drift velocity can be reduced, accelerated, or diflicted by an air blast. These measurements demonstrated that our computed drift velocities were correct.

One surprising result was the low light emision of the air plasma. Recombination apparently does not produce visible radiation. Exposures to high speed film in the visible and infra-red of an hour produced no visible results. Attempts to see visible decay

using a spark-gap switch to turn off the discharge and a photomultiplier to view the discharge gave no repeatable results. Thus, we can produce dense atmospheric plasmas in air, but cannot see them.

New Results

We have produced atmospheric plasmas in Argon and Helium, with dramatically different results. Using a plastic bucket to contain the gas, and a simple plastic sheet snapped over the container to provide a gas-air interface, we produce plasmas that are over 10^4 times as luminous as with air. We produce breakdown in atmospheric pressure at extremely low voltages and large distances. Several industrial applications are derived from the above results.

Software Development

We have had good progress on the development of a GUI for the AFOSR PIC code users group. Software development has progressed to the point where on screen grids, conductors, and dielectric boundaries are displayed, and some input lines are written. Appendix 1 is a compressed diskette of the Unix and IBM source code to date. Appendix 2 contains a report by Roger Horn on the work done on this project.

Patent Disclosures

The use of ionization in air to remove particulate matter and pollutants is quite well-known. We are working with the Southern

Research Corporation and PEAC on this matter. Recently, we have found a way of dramatically increasing the ionizing output. By using a non-electro-negative gas feed around the ionizing electrode, the production of ions is greatly enhanced. This idea is not new. However, what is new is that this local ion productioin extends out of the local gas cloud to produce enhanced ion density in air. In other words, the ions produced in a small, localized gas cloud can survive the air-gas interface and be useful over large volumes.

We have had a student project by Jim Muzzal under the direction of professor Alexeff, plus extra experiments by Scott Freeland that demonstrate the effect. In one experiment, argon gas was fed over the electrode of a Tesla Coil. The result was a dramatic plasma "flame". Other experiments showed the current enhancement under less dramatic conditions. These results are documented and photographed.

Appendix 1

Appendix 2

1 Summary

We developed a MAGIC graphical user interface program that can be used to define the element configuration for a MAGIC simulation problem. The interface program, MAGUI (for **MAGIC Graphical User Interface**), allows the configuration of conductor, dielectric, and look-back elements to be specified graphically. This graphical method of defining the structure shape has obivous advantages over the standard method of describing the configuration by point coordinates in a MAGIC simulation source file.

1.1 Objectives

MAGUI can be compiled with drivers for many different computing platforms and is presently able to run under both MS-DOS and X-windows environment. The basic graphics routines used in MAGUI are contained in a library developed at the University of Melbourne. The graphics library, VOGLE (for **Very Ordinay Graphics Learning Environment**) is a C routine library for doing line, polygon, arc, and circle drawings and fills. A major feature of the VOGLE graphics library is that low level driver routines for many different computing platforms are included in the library. By using the VOGLE library, porting the MAGUI code to another platform within the library is a relatively simple task.

1.2 Supported computing platforms

MAGUI program development was done on a Sun workstation under X11R5, and the program has been tested on an IBM PS/2 Model 50Z, AT 386SX and AT 386DX compatibles with SVGA video cards. The program code compiles under GNU gcc and Turbo C++, version 3.0.

Drivers for the following graphics systems are contained in the VOGLE library:

- IBM PS/2 and AT compatibles under MSDOS or PCDOS with one of the following video cards,
  - VGA
  - EGA
  - CGA
  - Sigma
  - Hercules Monochrome
- X-Windows (X11), release R2, R3, R4, and R5
- Microsoft Windows, version 3.x
- Sun Workstation
- Apollo Workstation
- Tektronix (401x)

- Postscript
- HPGL
- DXY

2 MAGUI Program Functionality

MAGUI consists of several screens, or *pages*, that are selected from the main menu. The main menu is always visible, and changing from one page to another can be done at any time. There are currently two pages in the available in the program: the **Layout** page and the **Files** page.

The Layout page contains the drawing board and is used to define the MAGIC simulation problem. The Files page is a dialog screen that is used to specify the input and output file names for MAGUI file I/O. The present version includes only routines for writing the problem configuration. Future versions will implement routines for reading a problem configuration from a MAGIC source file.

2.1 The Layout page

The Layout page (Figure 1) is the primary screen in the program. Contained on this page is the layout grid which may be a rectangular or polar grid depending on the selection of the coordinate system. The page menu is in the upper right. Below the page menu are the **Coordinate System**, **Grid Type**, and **Grid Size** popup window buttons.

Figure 1 shows a rectangular grid system that is used with the CARTESIAN (x, y, z) and CYLINDER-THETA (z, r, θ) coordinate systems in MAGIC. Figure 2 shows a polar grid that is used with the CYLINDER-Z (r, θ, z) coordinate system. The present version of MAGUI does not support the SPHERICAL (r, θ, ϕ) coordinate system.

Drawing a problem configuration on the Layout grid is accomplished with the console pointing device, which is typically a mouse. MAGUI may be used with either a 2-button or 3-button mouse. Currently, the left mouse button is used for all drawing and selecting functions. Defining a MAGIC simulation configuration requires drawing the conductor and dielectric elements. Drawing mode selection for the Layout page is done using the selector buttons in the **Drawing Controls** window. The buttons labeled **Conduct**, **Dielect**, and **Lookbak** select the element type that is to be placed on the drawing board.

The following actions place an element segment on the drawing board;

1. position the mouse cursor at one end of the element segment,
2. press and hold the left mouse button ("cross-hairs" are displayed at the segment end position),
3. position the mouse cursor at the other end of the segment, and
4. release the mouse button.

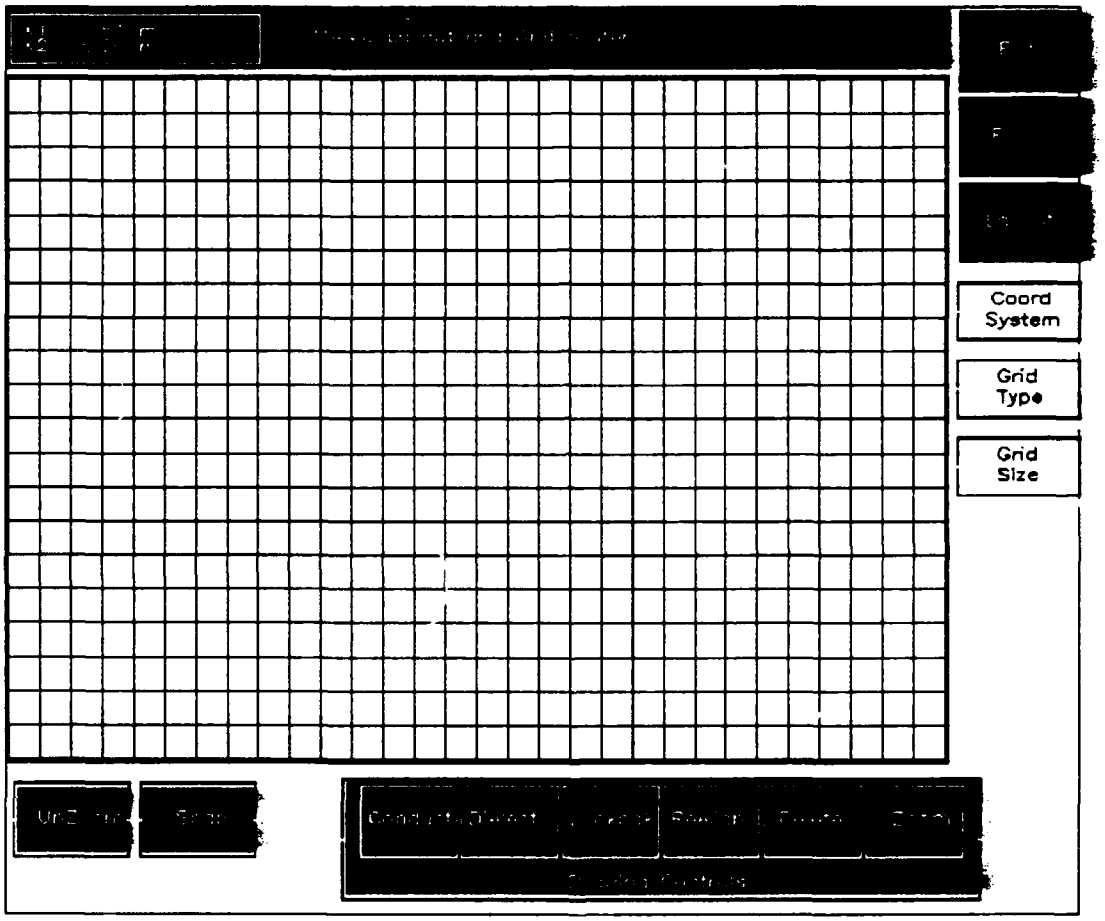

Figure 1: Layout page showing a rectangular grid system, drawing controls selector, and coordiate and grid popup buttons.

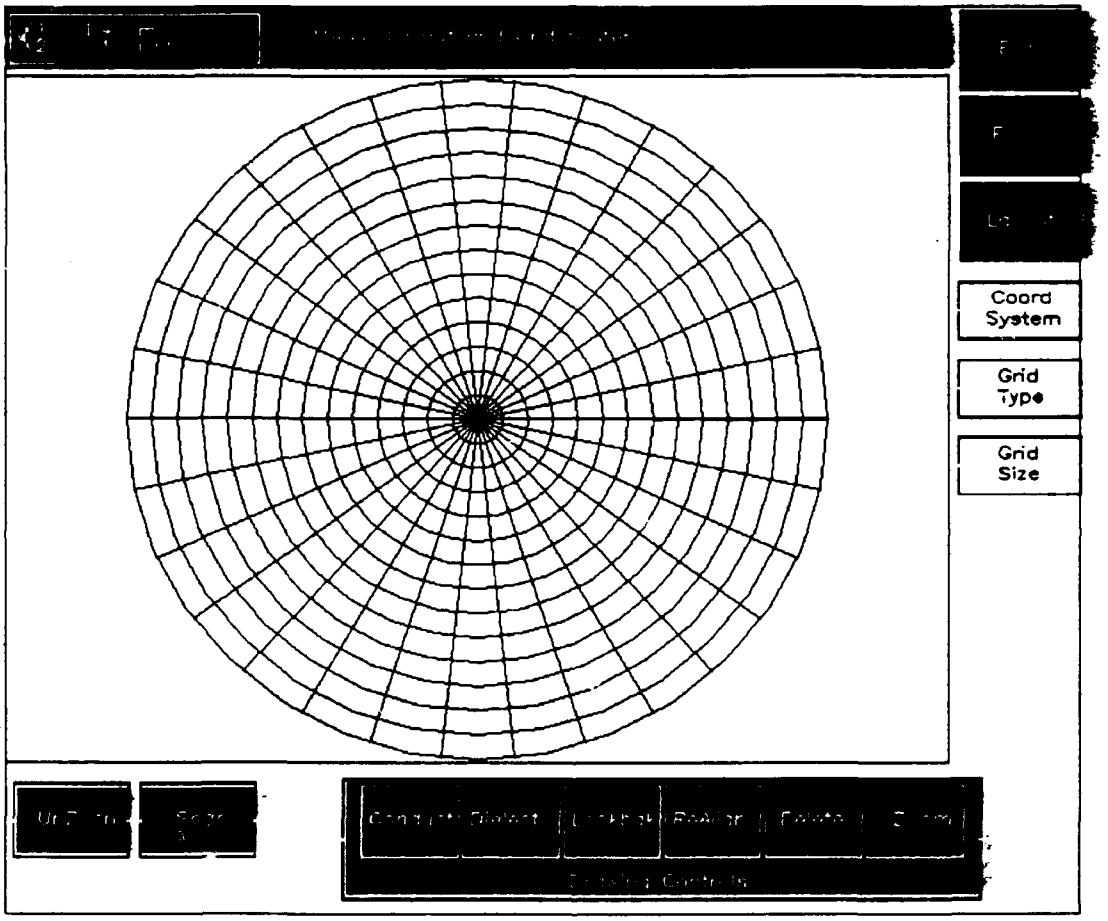

Figure 2: Layout page showing a polar grid system.

The new element segment is now displayed on the grid system.

A **Snap** mode selector button is displayed next to the **Drawing Controls** and is used to toggle between the snapped and unsnapped display modes. In the snapped display mode the element segment end points are "snapped" to the nearest coordinate points before the segment is displayed on the drawing board. In the unsnapped displayed mode the segments are displayed exactly as positioned and are not aligned with the coordinate system grid points.

The initial alignment assignment for CONDUCTOR and LOOKBACK elements on the drawing board is ALIGNED. The **ReAlign** button allows an element's alignment to be changed to ANTI-ALIGN. To change the alignment, the mouse cursor is positioned on the element, and the left mouse button is clicked once. The **ReAlign** mode is implemented as a toggle function, so each mouse click on an element toggles the alignment between ALIGN and ANTI-ALIGN.

The **Delete** mode is used to delete elements in the layout. The mouse cursor is positioned on an element, the left mouse button is clicked, and the element is removed from the layout. (Note: ReAlign and Delete modes are not implemented yet.)

A **Zoom** mode is included in the layout drawing control functions and allows any rectangular region on the current drawing board to be expanded to fill the drawing board. To use the **Zoom** mode, the mouse cursor is positioned at one corner of the desired region, and the left mouse button is clicked and held. The mouse cursor is then positioned on the opposite corner of the region, and the mouse button is released. Figure 3 shows a region of the drawing board depicted in Figure 2 that has been expanded using the **Zoom** mode. The **UnZoom** button returns the drawing board to the original scale after one or more **Zoom** operations.

The **Coordinate System** button produces a popup window that displays the **Select Coordinate System** menu (Figure 4). The four buttons are, **Cart**, **Theta**, **Z**, and **Spher** and specify SYSTEM selections of CARTESIAN, CYLINDER-THETA, CYLINDER-Z, and SPHERICAL, respectively. CARTESIAN and CYLINDER-THETA systems are displayed on a rectangular grid, and a CYLINDER-Z system is displayed on a polar grid. SPHERICAL systems are not supported in the current version of MAGUI.

The **Grid Type** button causes a popup window to be displayed that shows the **Select Grid Type** menu (Figure 5). Three buttons, labeled **Unif**, **Func**, and **Expl**, allow the selection of grid types UNIFORM, FUNCTION, and EXPLICIT. Presently, only the UNIFROM grid type is supported.

The **Grid Size** button displays a popup window that enables setting the number of grid points for the X1GRID and X2GRID (Figure 6). When acitvated, the window shows two sliding-bar selectors. The number of grid points is varied by positioning the mouse cursor on the sliding-bar for the desired grid. The left mouse button is clicked and held as the bar is moved. Moving the bar up increases the number of grid points. The minimum number of grid points that may be selected is 2 and the maximum is 102. The grid system is recalculated and displayed after the **Exit** panel in the popup window is selected with the mouse.

2.2 The Files page

The Files page (Figure 7) is used to specify an input file name, for reading a MAGIC source file, and an output file name, for writing MAGUI-generated, MAGIC source code. To enter an input

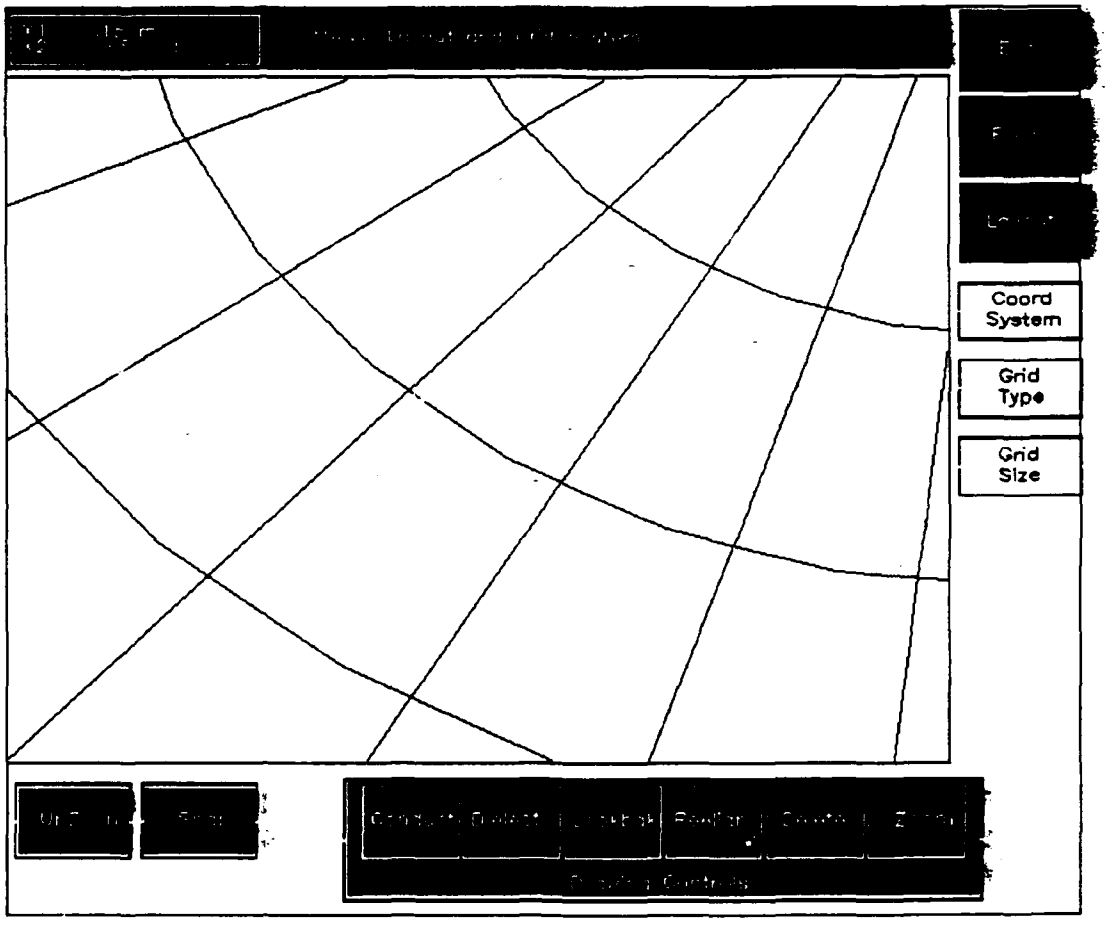

Figure 3: Region of CYLINDER-Z grid expanded using the Zoom mode.

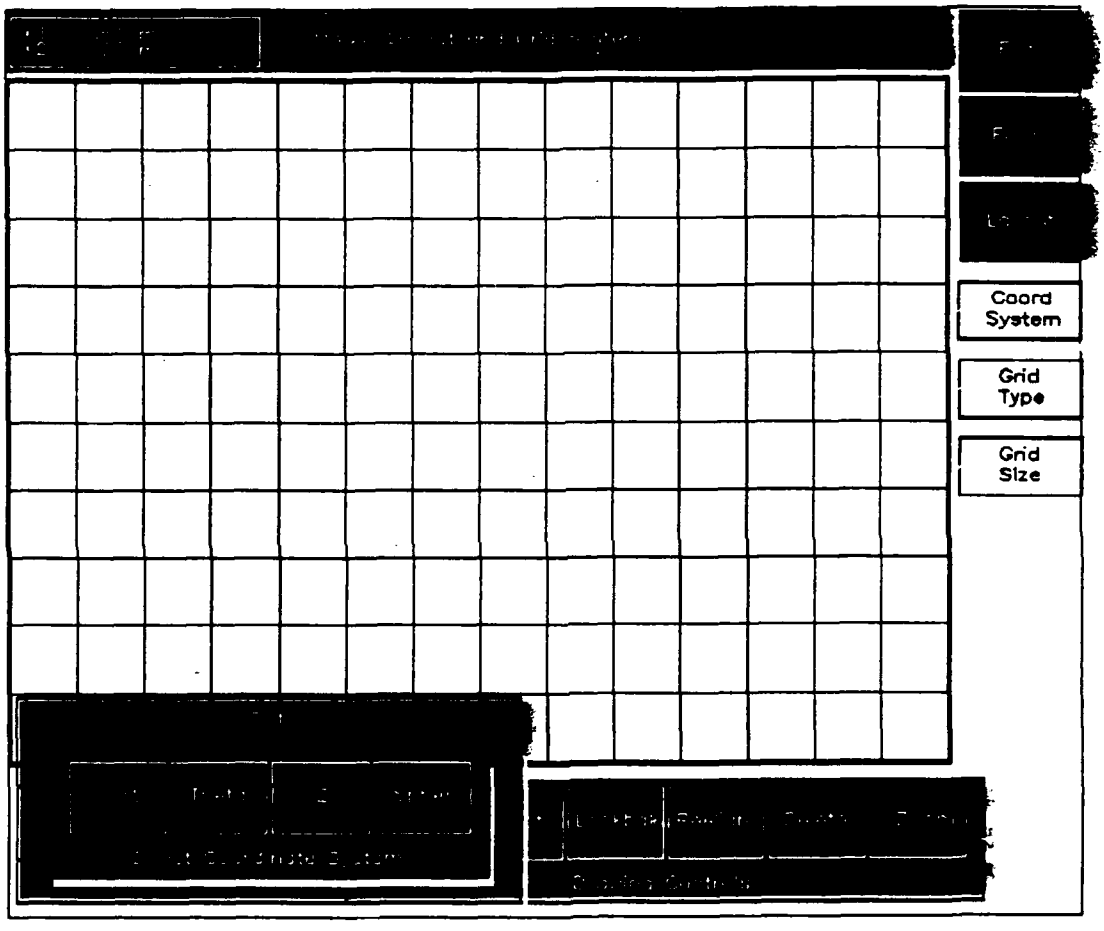

Figure 4: Layout page with the Coordinate System popup window activated.

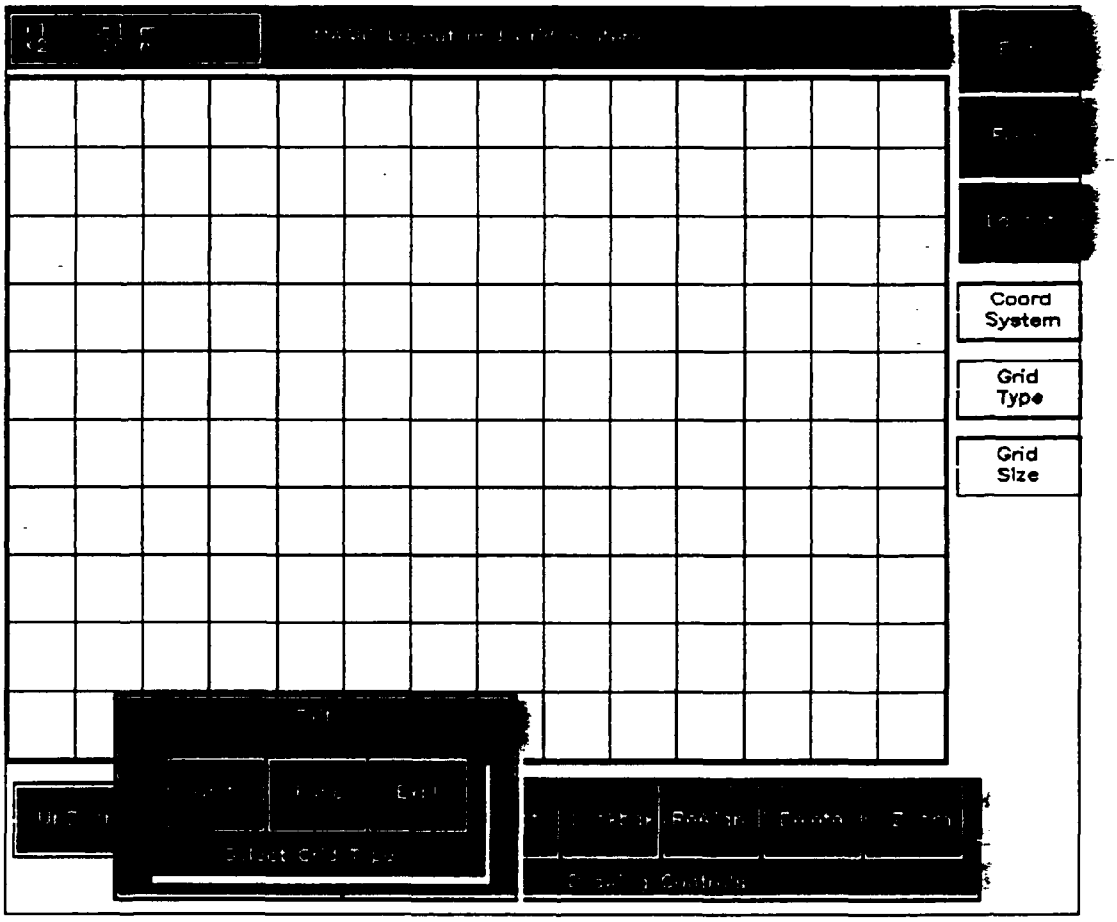

Figure 5: Layout page with the Grid Type popup window activated.

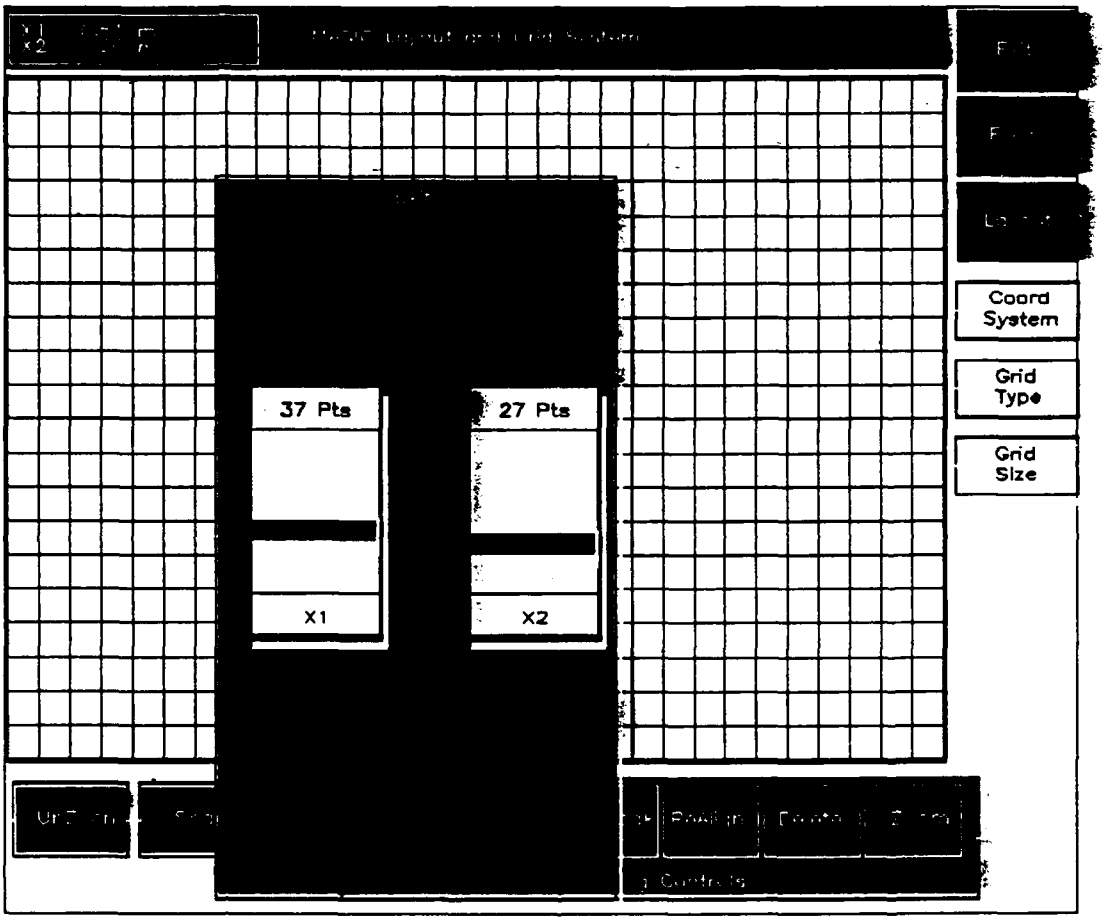

Figure 6: Layout page with the Grid Size popup window activated.

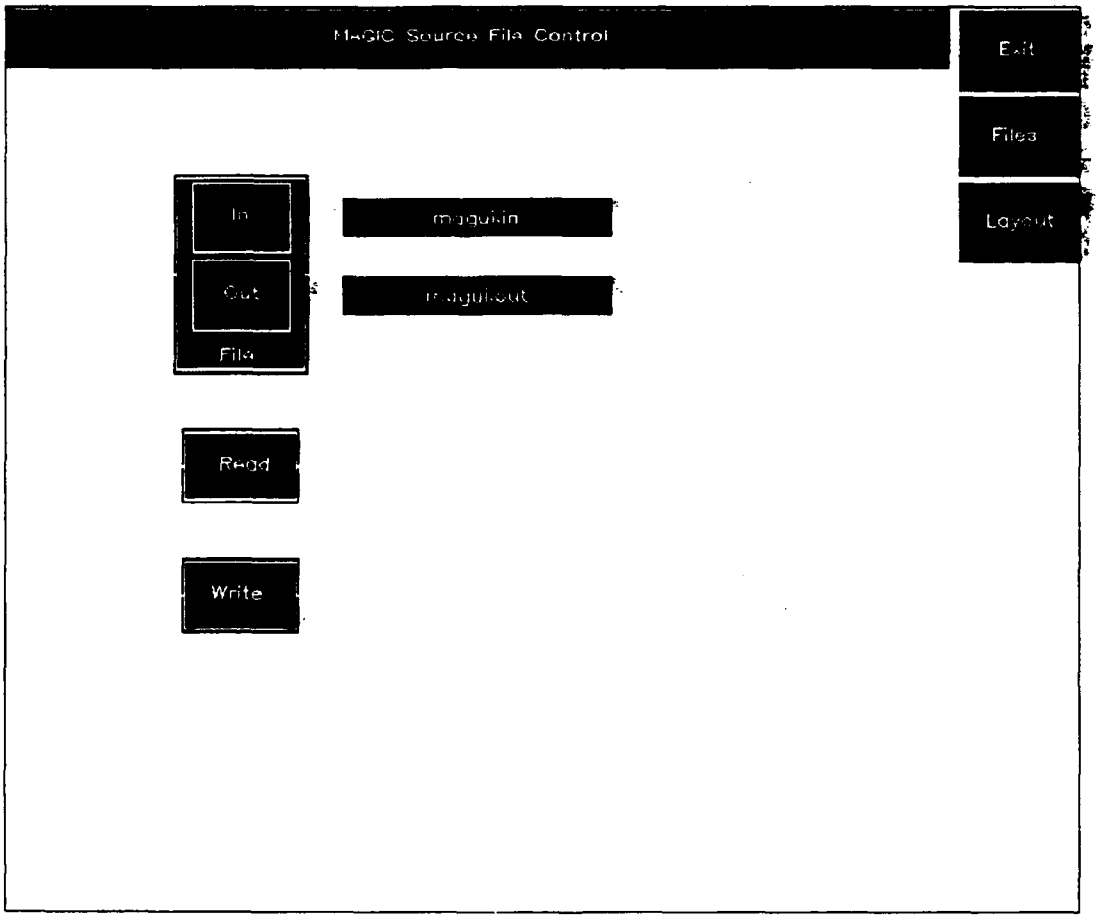

Figure 7:

file name, the **In** button in the **File** window is selected with the mouse, and a new input file name is entered. To enter an output file name, the **Out** button is selected, and a new output file name is entered. MAGUI file I/O is accomplished by clicking the **Read** or **Write** buttons.

3 MAGUI Program Code

The program routines and modules that comprise the MAGUI code are partitioned into three subsystems. These subsystems are:

- MAGUI page, selector, and button descriptions and coordinate grid system routines,
- basic graphical user interface routines,

- VOGLE graphic and driver routines.

The three subsystems provide a layered, hierarchical software design structure (Figure ??) for the program. The top layer is composed of the MAGUI specific routines which consist of display screen definitions and coordinate system class structures. The center layer consists of the graphical user interface modules and class structures and contains the basic building blocks for the display screen elements such as legends, push buttons, popup windows and selector button arrays. The bottom layer contains the low level routines for drawing lines, polygons, circles, and graphical display of text information.

3.1 MAGUI routines

A major part of the MAGUI specific routines is the definition and description of display screens used in the program. These class objects require specification of the object parameters such as height, width, and position on the screen. There are two basic mechanisms for instantations of object classes; parent class construction and button activation. A distinguishing feature of the GUI class objects is that they are contained within one of the page class objects (Layout or Files), and the objects exist only when their associated parent page is active.

Another important function of the MAGUI routines is calculation, control, and display of the coordinate grid system. The coordinate grid routines are contained in a single global class for the the coordinate system. The coordinate system class object exists for the duration of the program.

3.2 Basic graphical user interface

The graphical user interface implemented in the MAGUI program closely follows the class structures describe by Faison [1] and contains components that are used to construct the display screens. These components can be grouped as follows:

- events, graphics, and screen handler routines
- simple derived class components
- compound derived class components

The implementation by Faison is specific for the Zortech C++ and Turbo C++ compilers and uses the graphics libraries supplied with these compilers. So that the program can execute on many different platforms, the original graphical user interface code has been modified to support the VOGLE graphical system.

3.3 VOGLE graphics and drivers

The VOGLE library consists of C routines for doing line drawings, polyfills, circles, arcs, and graphics mode text display in a device independent fashion. The routines are divided into two groups: 1) position, line, and arc calculation routines, and 2) device drivers for the supported computing platforms. The interface between the calculation routines and the device drivers is handled using a graphics-primitive token mechanicism.

3.4 Class hierarchiy

The following is a list of the base classes in the MAGUI program.

- These classes define the coordinate system.

  CoordSystem
  XGrid
  XGridFunction

- These classes store information about the conductor element position and type.

  Conductor
  ConductorList
  ConductorListNode

- These classes store information about the dielectric element position and type.

  Dielectric
  DielectricList
  DielectricListNode

- These classes store information about the Lookback Field element position and type.

  LookbackField
  LookbackFieldList
  LookbackFieldListNode

- This is a virtual class. Every graphical object is derived from this class.

  Icon

- These classes are initialization and direct user interface.

  Configuration
  Events
  Graphics
  Screen
  Mouse

- These are utility classes.

  DisplayList
  DisplayListNode
  Legend

The derived class hierarchy is shown in Figure 8.

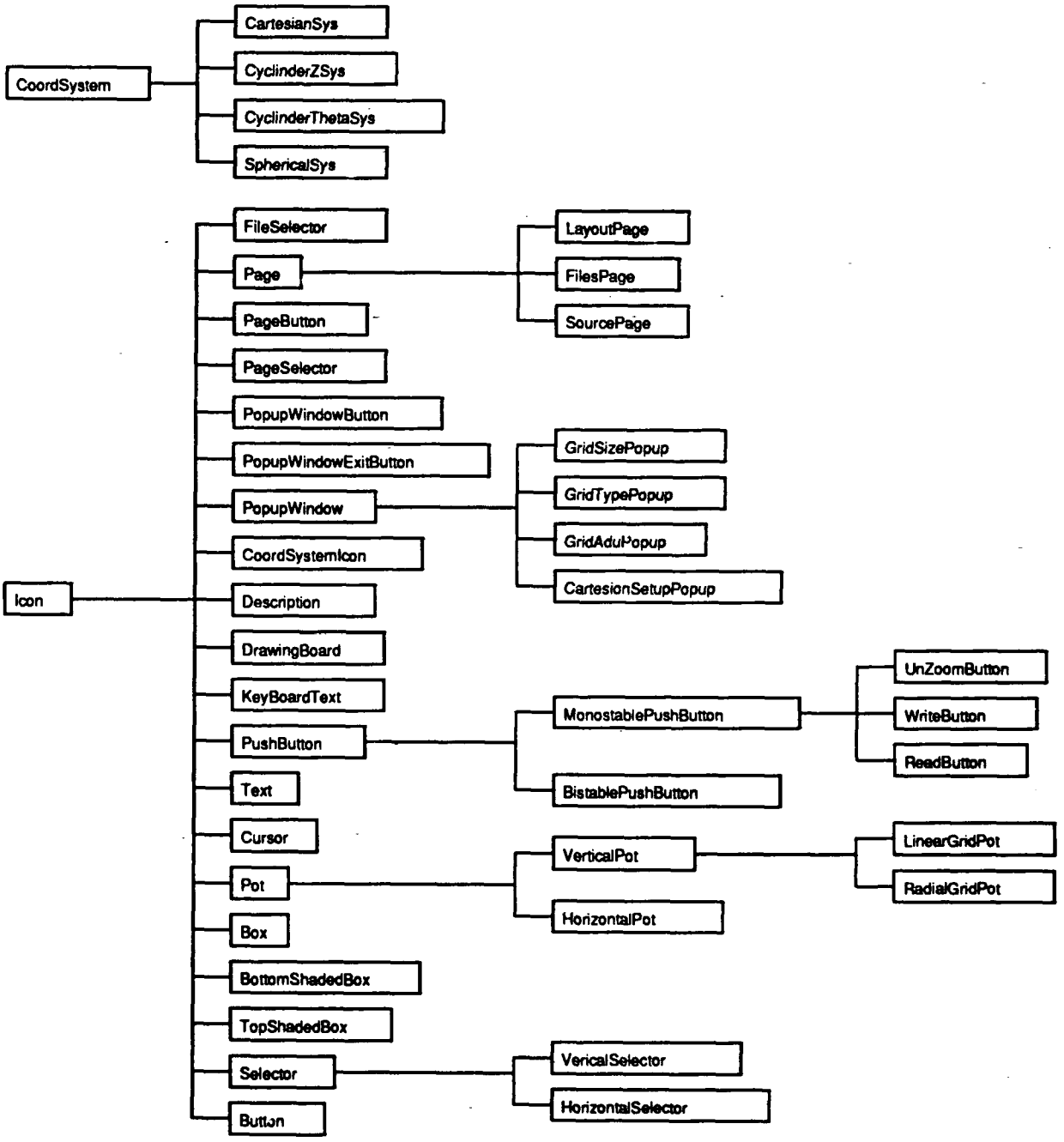

Figure 8: MAGUI class hierarchy.

4 Future Work

The present version of MAGUI is a first cut at an object-oriented graphical user interface for MAGIC, and future develpment would allow expansion of the basic program. A simple text file editor could be added to the program enabling the MAGIC programmer to work in either a graphical or textual mode. Adding an editor to MAGUI would bring it closer to an integrated development environment.

References

[1] Faison, E. W., Jr., *Graphical User Interfaces with Turbo C++*, SAMS, Macmillian, Inc., 1991.